Performance of Host Identity Protocol on Lightweight Hardware

Andrey Khurri
Helsinki Institute for Information Technology
Finland
akhurri@hiit.fi

Ekaterina Vorobyeva
Helsinki Institute for Information Technology
Finland
evorobyeva@yahoo.com

Andrei Gurtov
Helsinki Institute for Information Technology
Finland
gurtov@hiit.fi

ABSTRACT

The Host Identity Protocol (HIP) is being standardized by the IETF as a new solution for host mobility and multihoming in the Internet. HIP uses self-certifying public-private key pairs in combination with IPsec to authenticate hosts and protect user data. While there are three open-source HIP implementations, no experience is available with running HIP on lightweight hardware such as a PDA or a mobile phone. Limited computational power and battery lifetime of lightweight devices raises concerns if HIP can be used there at all. This paper presents performance measurements of HIP over WLAN on Nokia 770 Internet Tablet. It also provides comprehensive analysis of the results and makes suggestions on HIP suitability for lightweight clients.

Categories and Subject Descriptors

B.8 [**Performance and Reliability**]: Miscellaneous; C.2 [**Computer-Communication Networks**]: Network Protocols

General Terms

Measurement, Performance, Experimentation, Security

Keywords

HIP, Internet Tablet, PDA, public key signature, encryption, Diffie-Hellman key exchange, mobility

1. INTRODUCTION

The current trend of moving mobile telecommunication systems to IP technology is well-recognized. However, the security aspect of using IP protocol stack on lightweight devices, such as PDAs or mobile phones, is not sufficiently explored. In particular, encryption and public key signatures implemented in software are computationally expensive operations that could stress CPU and battery resources of mobile devices. The data throughput and latency can be negatively affected as well.

MobiArch'07, August 27–31, 2007, Kyoto, Japan.

The Host Identity Protocol (HIP) is a new secure mobility protocol specified by the IETF [7, 8, 5, 10, 9, 2]. HIP uses IPsec encapsulation for data packets and a version of Diffie-Hellman protocol to exchange public keys of two hosts. In this paper, we describe performance measurements of our port of HIP for Linux (HIPL) implementation to the Nokia 770 Internet Tablet, a Linux-based PDA. Although several previous projects evaluated HIP on standard Internet hosts [4, 3, 11, 6], none has targeted a HIP assessment on a mobile device with restricted resources. To check whether running IP-based security on lightweight hardware is feasible, we performed HIP measurements over WLAN with Nokia 770 acting as a mobile client, thus mainly being the initiator of a HIP association. In particular, we measured data throughput, latency, and power consumption of the HIP base exchange and mobility update. We then analyzed the results and suggested conditions where unmodified HIP would be suitable for use on lightweight hardware.

The choice of Nokia 770 as a target device for our measurements had been supported by several factors. First of all, it is a PDA with limited resources providing a good example of lightweight hardware to test HIP on. Secondly, such a handheld ideally represents a mobile client constantly moving across the Internet. In this approach, the tablet would be a desired device for the HIP protocol to be deployed on to deal with mobility issues. Next, Nokia 770 is becoming more and more attractive for both users and developers resulting in a number of applications (VoIP, Audio, Video on Demand, etc.) that might utilize the benefits of HIP. Finally, since Nokia 770 is a Linux-based PDA it is easier for any software (including HIP) to be ported to the open source platform.

The rest of the paper is organized as follows. In Section 2, we give relevant background on the HIP protocol. Section 3 briefly describes the Nokia Tablet hardware and our port of HIPL implementation. Section 4 contains measurement results of the basic HIP characteristics and their in-depth analysis. Section 5 concludes the paper with a summary of results and plans for future work.

2. HOST IDENTITY PROTOCOL

The existing Internet architecture that had been primarily designed for stationary hosts nowadays faces many nontrivial challenges with the growing amount of mobile terminals. Currently, there are two name spaces that are used globally by the Internet services and applications, domain names and IP addresses. IP addresses serve the dual role in the Internet being both end host identifiers and topo-

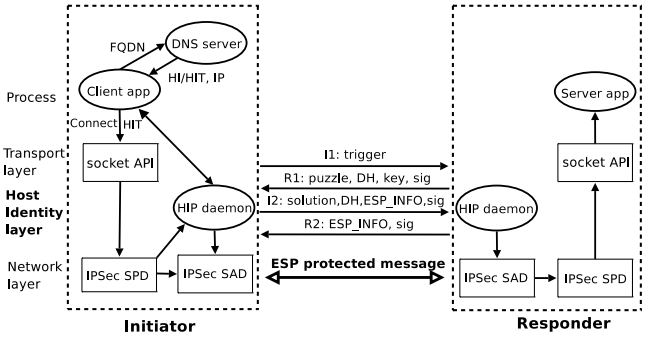

Figure 1: HIP Architecture.

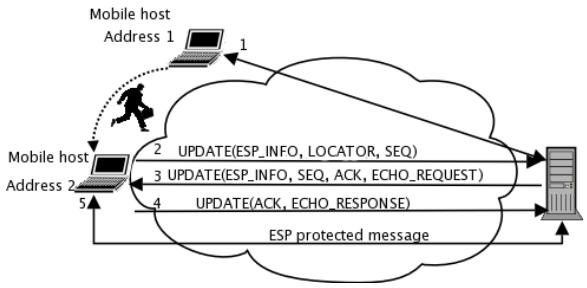

Figure 2: HIP Mobility Update.

logical locators. This general principle does not allow hosts to change their location without breaking ongoing transport protocol connections that are strictly bound to IP addresses.

2.1 HIP architecture

The Host Identity Protocol (HIP) [7] had been proposed to overcome the above mentioned problem. The idea behind HIP is decoupling the network layer from the higher layers in the protocol stack architecture (see Figure 1). HIP defines a new global name space, the Host Identity name space, thereby splitting the double meaning of the IP addresses. When HIP is used, upper layers do not any more rely on IP addresses as host names. Instead, Host Identifiers are used in the transport protocol headers for identifying hosts and establishing connections. IP addresses at the same time act purely as locators and are responsible for routing packets towards the destination. A Host Identifier is a public key of the host. For compatibility with IPv6 legacy applications, a Host Identifier is further represented by a 128-bit long cryptographic hash, the Host Identity Tag (HIT).

HIP offers several benefits including end-to-end security, resistance to CPU and memory exhausting denial-of-service (DoS) attacks, NAT traversal, mobility and multihoming support.

2.2 Base exchange

To start communicating through HIP, two entities must establish a HIP association. This process is known as the HIP Base Exchange (BE) [8] and it consist of four messages transferred between the initiator and the responder. After BE is successfully completed, both hosts are confident that private keys corresponding to Host Identifiers (public keys) are indeed possessed by their peers. Another purpose of the HIP base exchange is to create a pair of IPSec Encapsulated Security Payload (ESP) Security Associations (SAs), one for each direction. All subsequent traffic between communicating parts is protected by IPSec. A new IPSec ESP mode, Bound End-to-end Tunnel (BEET) [10] is used in HIP. The main advantage of BEET mode is low overhead in contrast to the regular tunnel mode.

Figure 1 illustrates the overall HIP architecture including the BE. The initiator may retrieve the HI/HIT of the responder from a DNS directory [9] by sending a FQDN in a DNS query. Instead of resolving the FQDN to an IP address, the DNS server replies with an HI (FQDN→HI). Transport layer creates a packet with the HI as the destination point identifier. During the next step, HI is mapped to an IP address by the HIP daemon on the Host Identity layer. Finally, the packet is processed by the network layer and delivered to the responder. As a result, the conventional 5-tuple socket becomes {protocol, source HI, source port, destination HI, destination port}.

2.3 Mobility and multihoming

Since neither transport layer connections nor security associations (SAs) created after the HIP base exchange are bound to IP addresses, a mobile client can change its IP address (upon moving, due to a DHCP lease or IPv6 router advertisement) and keep on transmitting ESP-protected packets to its peer. HIP supports such mobility events by implementing an end-to-end signaling mechanism [2] between communicating nodes (see Figure 2).

The purpose of the first UPDATE packet is to notify the peer of a new IP address and ESP information associated with this address. The corresponding parameters are called LOCATOR and ESP_INFO. The message also contains a SEQ parameter (a sequence number of the packet) and is therefore protected against possible losses by retransmission. Upon receiving the UPDATE message, the peer host must validate it, update any local HI↔IP mappings and assure that the mobile client is accessible via the new link. This is accomplished by sending the second UPDATE packet back to the mobile host at its new IP address containing an echo request along with the ESP_INFO of the peer. Finally, the mobile client is expected to acknowledge the message from its peer and return the content of the echo message. When the peer host gets this response, the new IP address of the client is marked as verified and the update procedure is completed. HIP multihoming uses same mechanisms as mobility for updating the peer with a current set of IP addresses of the host.

3. HIP ON NOKIA INTERNET TABLET

This section briefly introduces the Nokia 770 hardware and software, as well as lists requirements and steps taken to port HIP to it.

Nokia 770 Internet Tablet is a Linux-based handheld device with a high-resolution touch screen display, built-in WLAN, and Bluetooth support. Mainly designed for easy Web browsing, the tablet is also convenient for Internet telephony and instant messaging, reading emails and documents, playing media content. As its core, Nokia 770 has a Texas Instruments (TI) OMAP 1710 CPU running at 220 MHz. The device comes with 64 MB DDR RAM. The source of power for the tablet is a 1500 mAh Li-Polymer battery. The operating system is a modified version of Debian/GNU

Linux. For our experiments we used the latest release known as the Internet Tablet OS 2006 edition. It has a GNOME-based graphical user interface and runs the 2.6.16 Linux kernel.

Porting HIP to the Nokia 770 Internet Tablet consisted of a few steps. Since the handheld is running embedded Linux, we used an existing Linux implementation of the protocol, namely HIP for Linux (HIPL) developed at the Helsinki Institute for Information Technology. Although the HIP daemon and other utility programs of HIPL are userspace applications, a few modifications to the Linux kernel are necessary in order to support HIP. More specifically, three patches have to be applied to the Nokia kernel that must also have IPv6, IPsec, AES, 3DES, and SHA1 support.

Low computational power makes it impossible to compile large software projects as well as the Linux kernel directly on the PDA. For building HIPL userspace applications and the Nokia 770 Linux kernel we used a cross-compilation environment called Scratchbox. Located on a PC, Scratchbox toolkit emulates ARM environment and allows building the software for being used on the real devices. We then flashed our custom kernel image onto the device and packaged the userspace applications into a Debian binary file (*.deb) to be installed on the tablet.

4. EXPERIMENT RESULTS

This section presents the results of our experiments with the Host Identity Protocol on the Nokia 770 Internet Tablet. First, we introduce the platforms and the network environment we used. Then, in the following subsections we report measurement results and their interpretation.

4.1 Test environment

We performed our measurements on Nokia 770 Internet Tablet (Tablet) and Intel Pentium 4 CPU 3.00 GHz machine with 1 GB of RAM (PC) connected to each other via a switch and a WLAN access point (AP) in our test network. The network provided both IPv4 and IPv6 addresses. The wireless AP supported IEEE 802.11g standard and WPA (Wi-Fi Protected Access) encryption. All communicating parties used the same implementation of the HIP protocol. To better indicate the Tablet's performance level we repeated our measurement scenarios with a more powerful, 1.6 GHz IBM laptop (Laptop) connected to the PC over the same wireless link as the Tablet. Through a comparison we evaluated the impact of the Tablet's lightweight hardware on the maximum achievable data throughput, latency, duration of the base exchange and mobility update.

4.2 Basic HIP characteristics

4.2.1 Duration of HIP Base Exchange

A HIP association is set up by exchanging four control packets between communicating hosts. The purpose of measuring the HIP base exchange time was to determine the duration of various stages of BE such as generating and processing HIP messages by the Tablet. The measurement was performed using a script that established a HIP association 50 times in a number of scenarios. Since we did not find significant differences between IPv4 and IPv6 performance we present only results with RSA HITs mapped to IPv6 addresses of the hosts.

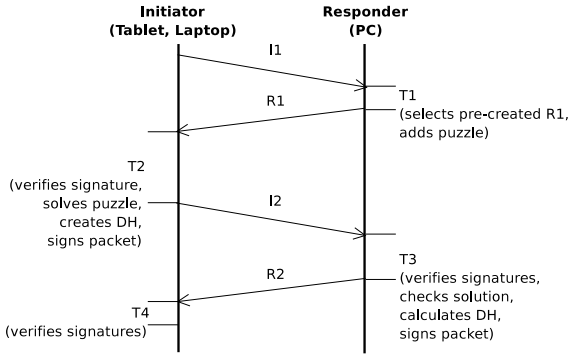

Figure 3: Time intervals measured on the Initiator and the Responder.

Figure 3 depicts the times that were measured in our experiments. We leave out I1 packet generation time due to its insignificance. T1 represents the time for the Responder to process an I1 packet and generate an R1. According to HIP implementation, Responder does not spend a lot of time for this phase since it chooses pre-created and signed R1 messages and adds a puzzle to them just before sending the packet to a network. The next time, T2, contains a number of CPU-intensive cryptographic operations such as generating and verifying signatures, calculating a Diffie-Hellman (DH) session key. During this stage the Initiator must also solve the challenge it received from the Responder. T3 indicates the time needed by the Responder to process an I2 packet that involves puzzle solution check, Initiator's public key verification and computation of the DH session key. If the puzzle was solved correctly, Responder generates an R2 message and signs it. Finally, during T4 the Initiator processes the R2 packet and completes the base exchange. At this point, the HIP association is established.

Figure 4 illustrates T1, T2, T3 and T4 times as well as the total duration of the HIP base exchange. We compare the results for two different HIP associations where the Initiators are Tablet and Laptop with the PC acting as the Responder. Thus, T1 and T3 times are measured for the PC whereas T2 and T4 times correspond to both Tablet and Laptop. As the figure indicates, the Laptop greatly outperforms the Tablet for all operations involved with BE. T2 time for the Tablet is nearly 1.2 seconds which is significantly longer than the respective one of the Laptop (0.14 seconds). The majority of T2 is spent by the Tablet on the operations with the public key signatures and generation of the Diffie-Hellman session key. This processing time heavily depends on the length of a public key and the DH Group ID. For our base tests on Tablet and Laptop we used the RSA key size of 1024 bits and 1536-bit DH Group.

The next test established a HIP association initiated by the PC while the Tablet acted as the Responder. Results suggest that the base exchange time is independent of whether PC or Tablet initiates the handshake. In both cases the base exchange lasts around 1.4 seconds.

Although 1.4 seconds to perform a HIP handshake between the Tablet and the correspondent PC might be acceptable for users and applications, HIP communication of two lightweight devices produces a higher delay. The duration of Base Exchange for a Tablet-to-Tablet scenario is over 2.6 seconds. Tablet spends a similar period of time in T2

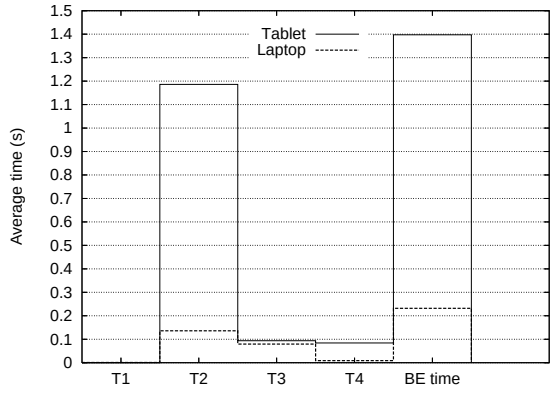

Figure 4: Duration of HIP Base Exchange stages for Tablet and Laptop.

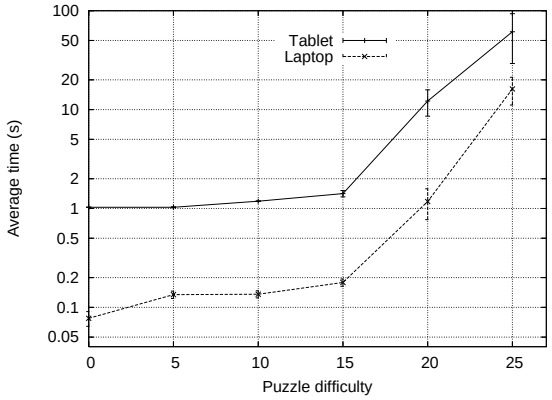

Figure 5: T2 processing time vs. puzzle difficulty.

and T3 phases. The amount of work by the Tablet-Initiator during the phase T2 is analogous to that performed by the Tablet-Responder during the phase T3. The only difference is that in T2 the Initiator spends the time for solving a cryptographic challenge whereas in T3 the Responder is supposed to verify the solution to that challenge and also validate the Initiator's HMAC signature. Otherwise, the same operations on public key signatures and Diffie-Hellman keys are carried out by both parties. Later, in the next section we will show that solving a puzzle with difficulty of ten makes a minimal impact on the T2 processing time. Considering this fact and also that puzzle solution check and HMAC validation in T3 are not computationally expensive we believe that the major influence on the BE parts and the total BE time is exerted by cryptographic operations costly for Tablet's CPU. Such operations include signatures verification and generation, as well as computation of the Diffie-Hellman session key.

4.2.2 Puzzle difficulty

Upon receiving an R1 packet, the Initiator is expected to solve a cookie challenge (puzzle) it gets from the Responder. This is done to protect the Responder against possible Denial-of-Service attacks by compelling the Initiator to spend a certain amount of CPU cycles to find a right answer. Depending on the conditions, i.e., on the trust level between the communicating endpoints, Responder has an opportunity to adjust the puzzle difficulty to be solved by the Initiator [8]. The difficulty (K) is represented by a number of bits that must match in a hash output sent back to the Responder. In the presented scenarios the default puzzle difficulty of ten was used. To see how the duration of the base exchange is affected by the puzzle difficulty we measured the time T2 with varying value of K. Figure 5 illustrates this dependency for the Tablet and the Laptop and shows that the time needed to solve the puzzle grows exponentially with increasing its difficulty. An interesting point we observed here is that the processing time starts rising dramatically when the puzzle difficulty is set to 15. Prior to this value the effect of increasing the difficulty level is tiny. There is a little difference between the processing times measured for the K values of zero and ten as compared to the T2 value itself of approximately 1 second. This consequently means a minor influence of the puzzle solving time to the total BE duration in our measurements with the puzzle difficulty of ten.

There is a time limit during which the Initiator must find a solution to the challenge. With Nokia 770, setting a high value of K by Responder would not be possible since Tablet's CPU will spend a long time to solve such puzzle. For example, a puzzle difficulty of 20 would keep Tablet's CPU busy for over 10 seconds which is unacceptable for most applications and users. The Laptop, in contrast, would solve a similar challenge in 1.3 seconds. Balancing between the puzzle difficulty and the time limit during which a correct solution is valid for the Responder might be an issue when using the lightweight hardware in a hostile environment with a low level of trust.

4.2.3 Diffie-Hellman

The Diffie-Hellman (DH) key exchange protocol is used in HIP to exchange the public keys of the hosts and produce a session key for the Initiator and the Responder. A piece of keying material is then generated from the session key and is used to create the corresponding HIP associations by the communicating parties [8]. The Responder includes in the R1 packet one or two its public DH keys. Upon receiving the R1 message with two DH values, the Initiator is supposed to select one that corresponds to the strongest DH Group ID it supports. Using different DH Groups makes it possible to affect the generation time of the DH session key and as a result the total duration of the HIP base exchange. In reality, this means an opportunity for a server to offer smaller DH public values to lightweight clients that are not powerful enough or if the security is not of critical importance.

We measured the T2 processing time containing the generation of the DH session key by the Initiator-Tablet and the Initiator-Laptop (see Figure 6). The graph shows an exponential growth in the processing time as the DH group ID increases. When using the weakest 384-bit DH Group, the Tablet is able to complete the T2 phase in less than 130 ms. This reduces the four-way base exchange to 200-300 ms with a PC as the Responder. With the 768-bit DH Group, T2 processing time for the Tablet is slightly higher and amounts to 234 ms. The total duration of the HIP BE with the PC in this case is about 340 ms. However, switching to the 1536-bit DH Group for better security, produces a longer delay close to 1 second. Further increasing the DH modulus length to 3072 and 6144 bits is not feasible for the Tablet

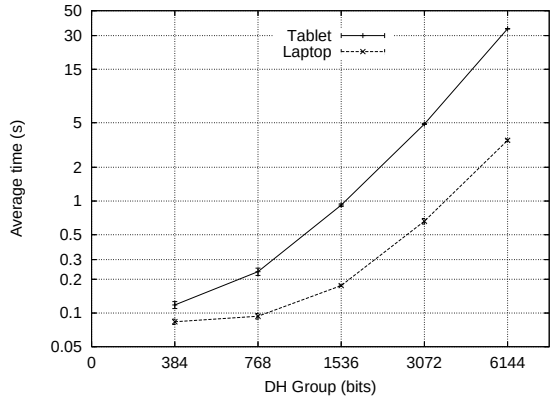

Figure 6: T2 processing time with different DH groups.

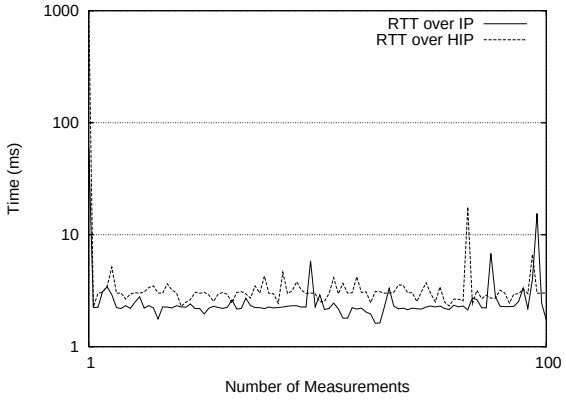

Figure 7: Round Trip Time, PC as the Initiator.

Table 1: Average Round Trip Times for Tablet and Laptop.

RTT	**Mean ± Standard deviation (ms)**		
	IPv6 (64B)	IPv6 (116B)	IPv6/HIP
PC→Tablet	2.223±0.470	2.358±0.425	2.936±0.931
Tablet→PC	1.901±0.332	1.900±1.235	2.748±1.347
PC→Laptop	1.026±0.340	1.049±0.312	1.177±0.243
Laptop→PC	1.065±0.338	1.070±0.427	1.207±0.502

as it results in the tremendous delays for the applications (over 5 and 35 sec correspondingly). In comparison with the Tablet, the Laptop is capable of handling the stronger encryption and spends less than 0.66 sec to compute the session key with the 3072-bit DH Group.

Our Diffie-Hellman measurements were conducted with a HIPL code snapshot as of May 2007 running on the latest version of the operating system on the Tablet. The DH experiment was also performed in a test network different from the one used for the rest of our HIP measurements. We see these factors as a reason for a difference of the results in the processing time of the HIP control packets on the Tablet (see, for example, Figure 5 and Figure 6).

4.2.4 RTT

The RTT (Round Trip Time) equals the time for a packet to travel from a node across a network to another node and back. Our tests evaluate the effect of the HIP protocol on RTT. Tests used the ping6 tool for sending 100 ICMP messages over HIP and over plain IP. We measured RTT for a number of scenarios including Tablet, Laptop and PC as HIP hosts.

Table 4.2.4 contains mean values of the RTT as well as standard deviations measured over IPv6 and over HIP. The first RTT value in each test is large because of the HIP base exchange and an ARP query performed upon the first connection. It is excluded from the average RTT calculations presented in the table. Figure 7 shows the whole set of RTT values with the PC acting as the Initiator. On average, HIP rises the latency by 35-45% as compared to the plain ICMP traffic.

The RTT time that we measured includes the transmission time of an ICMP ECHO_REQUEST message from the PC to the Tablet, processing time on both hosts and the latency of delivering an ICMP ECHO_RESPONSE back to the PC. The default size for an ICMP message equals 64 bytes (56 data bytes and 8 bytes of the ICMP header). When used with HIP, the size of an ICMP message is augmented by ESP headers and amounts 116 bytes. We measured the RTT time for a plain ICMP message of the size 64 bytes and 116 bytes as well as for an ESP encapsulated ICMP packet (IPv6 over HIP case). The results indicate that increasing the size of the ICMP packets affects the transmission latency by a small factor (6%). The major impact on the RTT over HIP (2.936 ms) is therefore made by a slow processing of the ICMP messages encapsulated with ESP. HIP increases the RTT value for the PC-to-Tablet connections on average by 37%. In contrast, the same proportion for the PC-to-Laptop scenario is around 15%. According to this comparison ESP encapsulation of the data involved by the Host Identity Protocol affects more seriously the lightweight devices than ordinary PCs or laptops.

4.2.5 Throughput

IPSec ESP data encryption performed by the Tablet can reduce the maximum achievable throughput over the wireless link. We measured TCP throughput by an iperf tool generating TCP packets to a correspondent node. It is necessary to mention that the WLAN Access Point introduces its own data encryption by means of the WPA protocol. Different tests had been performed to evaluate the overhead of ESP and WPA data encryption. The average values of the throughput are presented in the Table 2. An average value of 4.86 Mbps represents an upper bound of the throughput achievable by the Tablet acted as the initiator (see Tablet-to-PC scenario). This value was measured with a plain TCP/IP traffic in a totally open network with no encryption algorithms employed. Although the Tablet's specification claims supporting IEEE 802.11b/g standard with a maximum data rate of 54 Mbps, Tablet's CPU or possibly bad device driver implementation impose their own constraints. Further analyzing the results, we might conclude that WPA encryption makes a minor impact on the throughput. Enabling the WPA access control on the WLAN AP reduces the data rate only by 0.4% (4.84 Mbps vs 4.86 Mbps). In

Table 2: TCP throughput in different scenarios.

Throughput	Mean ± St.dev. (Mbps)	
	Tablet→PC	Laptop→PC
TCP	4.86±0.28	21.77±0.23
TCP/HIP	3.27±0.08	21.16±0.18
TCP+WPA	4.84±0.05	–
TCP/HIP+WPA	3.14±0.03	–

contrast, the ESP influence is much stronger and reduces the throughput by 32% (3.27 Mbps vs 4.86 Mbps) in the same network. Mutual impact of WPA and ESP is bigger as double encryption is used.

In comparison with Tablet, the Laptop achieves 21.77 Mbps of the TCP data rate over the same open wireless link (see Laptop-to-PC scenario). An interesting observation is that with Laptop the impact of ESP encryption involved by HIP is tiny as compared to Tablet and equals 3% of decrease in throughput.

Figure 8 graphically depicts the results and shows the distribution of TCP and TCP/HIP throughput over a WPA-free wireless link. The graph illustrates HIP influence on the TCP throughput as well as a difference in values achieved by the lightweight Tablet and a much more powerful Laptop.

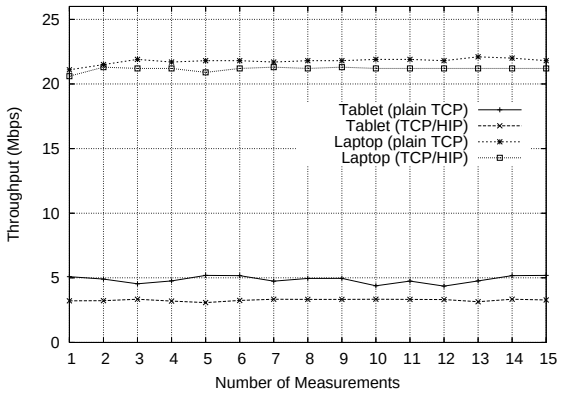

Figure 8: TCP throughput in an open wireless network.

End-to-end security provided by HIP might be used not only for data protection itself but also for authentication to an access point as an alternative to WPA algorithms in wireless networks. However, as the results above indicate for the devices with limited computational power, the data throughput and latency are significantly affected and might become a concern.

4.2.6 Duration of a Mobility Update

HIP sends Mobility Update packets when the IP address of a HIP mobile terminal changes. We measured the time to exchange Mobility Update packets by manually changing the IP address of the network interface to simulate a simple mobility case. We repeated our tests 35 times and calculated the average. The average duration of Mobility Update between the Tablet and the PC is 287 ms (see Figure 9). However, in reality this delay can be lower for applications. Once the correspondent node receives the first UPDATE packet it knows the Tablet's new location and can transmit data to the new address using Credit-Based Authorization (CBA). CBA limits the transmission rate to a new IP address until it is verified to be reachable by the last two UPDATE packets. Such practice prevents hijacking of arbitrary IP addresses. The average time for generating, sending and processing the first UPDATE packet is around 20 ms. Comparing to the Tablet, our 1.6 GHz laptop is capable of completing the three-way Mobility Update with its correspondent node in 100 ms.

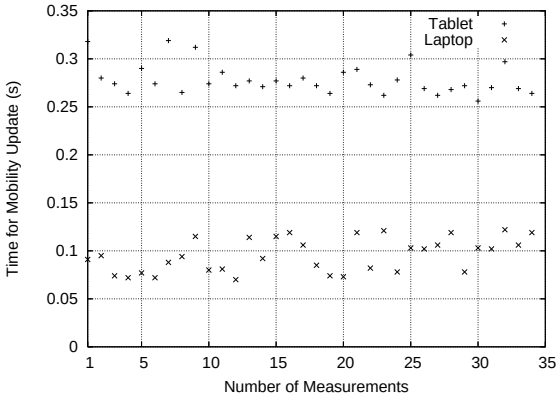

Figure 9: Duration of a Mobility Update.

4.2.7 Battery lifetime

Power consumption is a crucial issue for any portable device. The capacity of the Nokia 770's battery keeps the device in a standby mode for a few days. However, the battery resources are exhausted quickly by applications requiring data transmission over WLAN. The objective of measuring battery lifetime on the Tablet was to assess how expensive the Host Identity Protocol operations might be in terms of power consumption. We used an external multimeter to measure the consumption of the battery's current while the device was busy with various applications (see Table 3). Given the capacity of the battery and the current consumed by an application we were able to compute a theoretical time to deplete the battery. Alternatively, we ran the same application on the Tablet with a fully charged battery until its depletion to verify our empirical assumption about the lifetime. With HIP, the average current measured by the multimeter was 0.38 A. A fully charged 1500-mAh battery kept the Nokia Tablet working for about three and a half hours.

Our preliminary results show almost no difference in power consumption between the HIP-enabled and non-HIP applications. Establishing a HIP association, mobility update as well as ESP encrypted traffic all consume a similar amount of the current (0.36-0.38 A) equivalent to a plain TCP/IP data connection. We interpret these results as caused by the low computational power of the Nokia's CPU which is kept busy all the time upon transmitting data over WLAN regardless of the protocol and the application being used. We also believe that HIP does consume more power beside the non-HIP applications if compared to the data throughput. In other words, due to a lower bitrate caused by ESP data

Table 3: Power consumption by applications.

Application/Mode	Current (A)
HIP Base Exchange	0.36
ESP traffic (iperf with HIP)	0.38
Plain TCP (iperf without HIP)	0.38
Video stream from a server	> 0.50
Local video	0.27
Audio stream from a server	0.40 - 0.50
Local audio	0.20
Browsing (active WLAN)	0.35 - 0.50
Passive WLAN	0.12
Activating screen	0.12 - 0.14
Standby mode	< 0.01

encryption a HIP application would require a notably longer time for a similar task to be completed. For instance, Tablet is able to transmit 100 Mbytes of data in 170 seconds over plain TCP/IP while HIP would spend additional 98 seconds (and total time of 268 seconds) for the same piece of work. In terms of power consumption the use of HIP would therefore intend longer CPU utilization and consequently more energy consumed for a task.

5. CONCLUSIONS

This paper presented measurements and performance evaluation of Host Identity Protocol on Nokia 770 Internet Tablet. We found several interesting results on the use of asymmetric cryptography on lightweight devices.

- The unmodified HIP protocol may be used in scenarios where a lightweight device communicates through a single proxy server in the Internet. A HIP association establishment in such case is 1.4 sec and mobility update is 287 ms.
- For scenarios involving two mobile hosts or multiple parallel HIP associations, unmodified HIP is too heavy for lightweight devices. For two Tablets, the HIP association establishment is already 2.6 sec.
- For applications that do not require strong security (i.e., web browsing) the duration of the HIP association establishment with a server might be reduced up to 0.35 sec by using 768-bit DH Group in Diffie-Hellman key exchange.
- Surprisingly, the Tablet only achieves 4.86 Mbps in a WLAN capable of 22 Mbps even without HIP. The use of WPA encryption has negligible effect on throughput, but ESP encryption with HIP reduces the throughput to 3.27 Mbps. It is still sufficient for most Tablet applications.
- The RTT over WLAN is only several milliseconds. HIP increases the RTT by few milliseconds that does not noticeably affect the applications.
- The use of ESP encryption with HIP does not affect the battery consumption in the Tablet, although the energy cost per byte is higher with HIP due to reduced throughput. We noticed that the Tablet CPU is always fully utilized when an application transmits data over WLAN that depletes the battery in 3-4 hours.
- We believe that the measurements results are applicable to a wide range of mobility and security protocols in addition to HIP. Most such protocols rely on similar public-key and IPsec ESP operations like HIP.

Our measurements served as a motivation for proposing Lightweight HIP that uses hash chains instead of asymmetric cryptography [1]. Lightweight HIP achieves up to two orders of magnitude reduction of HIP computational cost at the expense of public key authentication. In future work, we plan to compare LHIP and HIP on newer Nokia N800 Tablet with video and VoIP capability, as well as evaluate the HIP implementation on Symbian OS platform.

6. ACKNOWLEDGMENTS

The authors thank Miika Komu, Tobias Heer, Tony Jokikyyny, Jarno Rajahalme, Petri Jokela, and anonymous reviewers for feedback that helped to improve the paper.

7. REFERENCES

[1] T. Heer. LHIP lightweight authentication extension for HIP: draft-heer-hip-lhip-00.txt, Feb. 2007. Work in progress.

[2] T. Henderson. End-host mobility and multihoming with the host identity protocol: draft-ietf-hip-mm-05, Mar. 2007. Work in progress. Expires in September, 2007.

[3] T. R. Henderson. Host mobility for IP networks: A comparison. *IEEE Network*, 17(6):18–26, Nov. 2003.

[4] T. R. Henderson, J. M. Ahrenholz, and J. H. Kim. Experience with the host identity protocol for secure host mobility and multihoming. In *Proc. of the IEEE Wireless Communications and Networking Conference (WCNC'03)*, Mar. 2003.

[5] P. Jokela, R. Moskowitz, and P. Nikander. Using ESP transport format with HIP: draft-ietf-hip-esp-03, June 2006. Work in progress.

[6] P. Jokela, T. Rinta-Aho, T. Jokikyyny, J. Wall, M. Kuparinen, H. Mahkonen, J. Melen, T. Kauppinen, and J. Korhonen. Handover performance with HIP and MIPv6. In *Proc. 1st International Symposium on Wireless Communication Systems, ISWCS'04*, Sept. 2004.

[7] R. Moskowitz and P. Nikander. Host identity protocol (HIP) architecture. RFC 4423, IETF, May 2006.

[8] R. Moskowitz, P. Nikander, P. Jokela, and T. R. Henderson. Host identity protocol: draft-ietf-hip-base-07, Feb. 2007. Work in progress. Expires in August, 2007.

[9] P. Nikander and J. Laganier. Host Identity Protocol (HIP) Domain Name System (DNS) extensions: draft-ietf-hip-dns-08.txt, Oct. 2006. Work in progress.

[10] P. Nikander and J. Melen. A bound end-to-end tunnel (BEET) mode for ESP: draft-nikander-esp-beet-mode-06, Aug. 2006. Work in progress.

[11] P. Nikander, J. Ylitalo, and J. Wall. Integrating security, mobility, and multi-homing in a HIP way. In *Proc. of Network and Distributed Systems Security Symposium (NDSS'03)*, San Diego, CA, USA, Feb. 2003. Internet Society.